DUMPSBOSS.

OMG-Certified Systems Modeling Professional - Model Builder – Intermediate

OMG OMG-OCSMP-MBI300

Version Demo

Total Demo Questions: 11

Total Premium Questions: 119

Buy Premium PDF

https://dumpsboss.co

support@dumpsboss.co

support@dumpsboss.co
dumpsboss.co

Topic Break Down

Topic	No. of Questions
Topic 1, Modeling Concepts	7
Topic 2, SysML Concepts	5
Topic 3, Requirements Modeling	10
Topic 4, Structural Modeling	30
Topic 5, Behavioral Modeling	45
Topic 6, Parametric Modeling	18
Topic 7, Cross-Cutting Relationships	4
Total	119

QUESTION NO: 1

Choose all that apply.

Which statements are true about control nodes in an activity diagram?

A. A decision node can route tokens according to guards on outgoing edges.

B. A merge node combines alternative flows without synchronizing them.

C. A join node can synchronize concurrent flows.

D. A fork node selects exactly one outgoing edge for each incoming token.

E. A merge node requires tokens from every incoming edge before it can offer a token.

ANSWER: A B C

Explanation:

A decision node routes incoming tokens to one of its outgoing edges according to guards. A merge node brings together alternative control-flow paths and does not synchronize them. A join node synchronizes concurrent flows by waiting for tokens according to its join specification before offering a token on its outgoing edge.

A fork node supports concurrent execution by offering tokens along its outgoing edges; it does not select only one outgoing path. These control nodes govern the routing and coordination of execution, rather than performing domain-specific actions. SysML reuses UML activity semantics; see the OMG UML 2.5.1 specification and the OMG SysML 1.6 specification.

QUESTION NO: 2

Choose the correct answer.

What is the purpose of an exception handler in an activity?

A. It specifies behavior to execute when an exception is produced by a protected node.

B. It synchronizes all concurrent flows entering an activity.

C. It specifies a guard for an ordinary decision node.

D. It prevents tokens from leaving an activity final node.

ANSWER: A

Explanation:

An exception handler associates a protected node with a handler body that executes when an exception is produced during execution of the protected node. The handler can accept the exception through an exception input and model the recovery, reporting, alternate processing, or other response required for that exceptional condition.

The exception handler does not represent an ordinary control-flow branch. It provides a structured means to specify behavior that is invoked as a consequence of an exception raised by the protected execution. SysML uses the UML activity semantics for exception handlers. See the OMG UML 2.5.1 specification and the OMG SysML 1.6 specification.

QUESTION NO: 3

Choose all that apply.

Which statements are true about transitions in a SysML state machine?

A. A trigger may identify an event that makes a transition eligible to fire.

B. A guard must evaluate to true before a guarded transition can be taken.

C. An effect behavior may be executed when a transition is taken.

D. A transition must always have both a trigger and a guard.

E. A transition may only originate from a final state.

ANSWER: A B C

Explanation:

A transition may have a trigger, a guard, and an effect. A trigger identifies an event that can make the transition eligible to fire. A guard is a Boolean expression that must evaluate to true for the transition to be taken. When a transition is taken, its effect behavior is executed as part of the transition.

A transition without a trigger is not necessarily taken immediately. It may be a completion transition, which becomes eligible when the source state completes its internal behavior, subject to any guard. SysML state machines use UML state-machine semantics; see the OMG UML 2.5.1 specification.

QUESTION NO: 4

Choose all that apply.

Which statements are true when modeling quantitative properties in SysML?

A. A value property can represent a quantity associated with a block.

B. A value type may specify a unit and a quantity kind.

C. A binding connector can bind a constraint parameter to a value property.

D. Every value property is implicitly read only.

E. A value property cannot be used in a parametric diagram.

ANSWER: A B C

Explanation:

A value property represents a quantity or value associated with a block, such as mass, temperature, capacity, or power. It can be typed by a value type. A value type may specify a unit and a quantity kind, providing a consistent interpretation of the values used in the model.

In a parametric context, a constraint parameter can be bound to a value property using a binding connector. The binding connector constrains the connected properties to have the same value. A value property is not inherently read-only; read-only is a separate property characteristic that may be specified when appropriate. See the value-type, value-property, and parametric-modeling definitions in the OMG SysML 1.6 specification.

QUESTION NO: 5

Choose the correct answer.

A system engineer has created a SysML block structure for his system-of-interest. Two of the blocks (M and E) represent system-level abstractions of components that will be designed by a mechanical engineer and an electrical engineer m their respective computer-aided design (CAD) tools.

Given that the CAD models are wrapped as blocks B_mcad and B_ecad, what type of diagram should the system engineer use to quantitatively relate the properties of B_mcad and B_ecad to their system level abstractions M and E respectively?

A. an activity diagram

B. a parametric diagram

C. an internal block diagram

D. a block definition diagram

ANSWER: B

Explanation:

A parametric diagram is the appropriate SysML diagram because it represents quantitative constraints and the binding of those constraints to value properties of blocks. In this situation, the wrapped CAD blocks, B_mcad and B_ecad, expose properties originating in the mechanical and electrical CAD models, while M and E provide the corresponding system-level abstractions. A parametric diagram can place constraint properties in a constraint context and connect their parameters to the relevant value properties through binding connectors. This makes the intended mathematical or engineering relationship explicit—for example, relating CAD-derived mass, dimensions, power, resistance, or other physical quantities to the values used by the system model.

SysML defines parametric diagrams specifically to support analysis of critical system parameters, including performance, reliability, and physical relationships. The diagram provides a model-based way to maintain traceable quantitative consistency between discipline-specific design representations and the system-level model. The constraints can also be evaluated by an analysis tool when suitable value bindings and constraint definitions are available. See the OMG SysML 1.6 specification, especially its definition of parametric diagrams and constraint properties, and the OMG SysML specification landing page.

QUESTION NO: 6

Choose all that apply.

Which statements correctly describe SysML requirement relationships?

A. A «satisfy» relationship is directed from a design element to a requirement.

B. A «verify» relationship can relate a test case to a requirement.

C. A «deriveReqt» relationship is directed from a derived requirement to its source requirement.

D. A «refine» relationship may be used only between two requirements.

E. A «verify» relationship indicates that the requirement text has been decomposed.

ANSWER: A B C

Explanation:

The «satisfy» dependency relates a design element, as client, to a requirement, as supplier, indicating that the design element satisfies the requirement. The «verify» dependency relates a verification element, such as a test case, as client to the requirement it verifies. The «deriveReqt» dependency is directed from the derived requirement to its source requirement.

The «refine» dependency is more general. It may relate a requirement to another model element that provides a more detailed or alternative representation of the requirement, such as an activity, use case, or other behavioral or structural model element. It is not restricted to relationships between requirements. See the requirements constructs in the OMG SysML 1.6 specification.

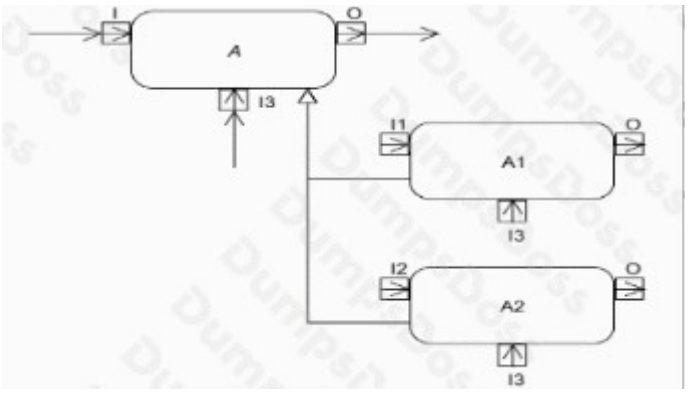
QUESTION NO: 7

Choose the correct answer

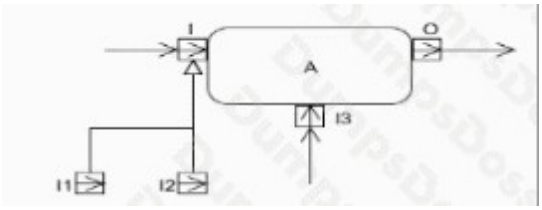
A has three inputs I 1. I 2 and 13 and one output O, I1, I2. I3 and O are all of the same type A requires I1 or I2 to execute but I1 and I2 are mutually exclusive. I3 must be present tor A to begin.

Which activity diagram fragment correctly shows A with its three inputs and one output?

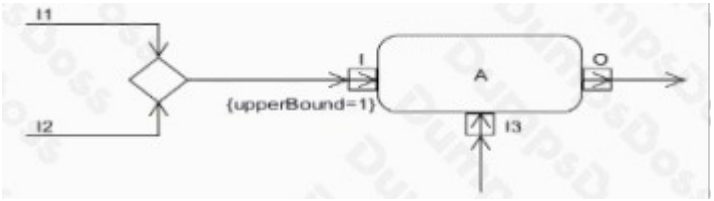
A)



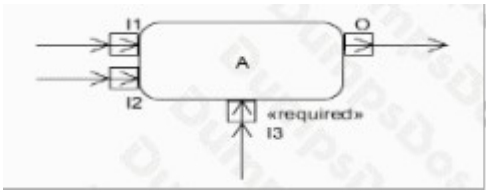
B)



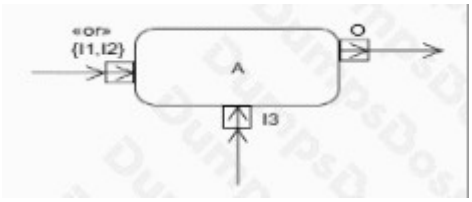
C)



D)



E)



A. Option A

B. Option B

C. Option C

D. Option D

E. Option E

ANSWER: A

Explanation:

Option A is correct because it models the required token semantics: the two mutually exclusive alternatives, I1 and I2, are brought together through a merge before the action is enabled in combination with the mandatory I3 input. A merge node accepts tokens from alternative incoming flows and passes whichever token arrives onward; it therefore represents the required “I1 or I2” condition without requiring both tokens. The separate path for I3 is synchronized with that merged alternative so that execution cannot proceed until an I3 token is also available. This captures the stated precondition precisely: one token of the common type from either I1 or I2, plus a token of that type from I3, is required before A begins. The output flow from A then carries the resulting token through O, which is also typed consistently with the inputs. In UML activity semantics, object flows carry tokens between object nodes and action pins, while merge and join control structures express alternative and required concurrent availability. The model fragment therefore uses the activity-diagram token model rather than merely showing three unrelated incoming flows. See the OMG UML 2.5.1 specification, especially its activity-node and object-flow semantics, and the OMG UML 2.5.1 overview.

QUESTION NO: 8

Choose all that apply.

Which statements are true about the SysML «allocate» relationship?

A. It is a specialized abstraction dependency.

B. It can allocate behavior to the structural element responsible for performing it.

C. It may be used to express a mapping between model elements.

D. It establishes composite ownership of the allocated element.

E. It is a replacement for a «satisfy» relationship to a requirement.

ANSWER: A B C

Explanation:

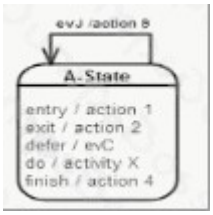
The «allocate» relationship is a specialized abstraction dependency used to represent an allocation of one model element to another. It is commonly used to relate behavior to structure, such as allocating an activity or action to the Block responsible for performing it. It can also express allocations between other kinds of model elements when the allocation meaning is defined by the modeling methodology.

The relationship does not assert that the allocated element is owned by, composed within, or necessarily satisfies the element to which it is allocated. Allocation provides traceability for responsibility or mapping decisions rather than replacing structural or requirements relationships. See the allocation definitions in the OMG SysML 1.6 specification.

QUESTION NO: 9

Choose the correct answer.

Assume the following state machine fragment Is In the A-State when the event evC arrives.



What is the order of behaviors that are executed after evj arrives?

A. action 9, action 1

B. action 2, action 9, action 1

C. action 2, action 9, action 1, start activity X, evC

D. stop activity X (if needed), action 2, action 9, action 1, start activity X

E. stop activity X (If needed), action 2, action 9, action 1, action 4 start activity X

ANSWER: D

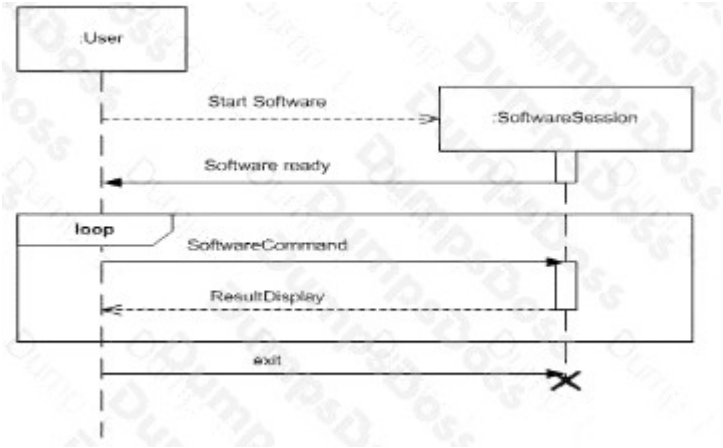
Explanation:

stop activity X (if needed), action 2, action 9, action 1, start activity X is correct because the event triggers an external transition that leaves and then re-enters the state owning activity X. UML state-machine execution requires an active doActivity to be terminated when its state is exited. The state's exit behavior, represented by action 2, then executes. Next, the transition effect, action 9, is performed. On reaching the target state, its entry behavior, action 1, executes, and its doActivity, activity X, is started again as part of entering that state. This order preserves the UML distinction between leaving a state, executing the transition's effect, and entering the target state. The wording "if needed" is appropriate because termination is relevant only if activity X is still executing at the moment the transition is processed. UML specifies these exit, transition-effect, entry, and doActivity lifecycle semantics for state-machine execution. See the OMG UML 2.5.1 specification, particularly its state-machine semantics, and the OMG SysML 1.6 specification, which uses UML state-machine behavior as its foundation.

QUESTION NO: 10 - (SIMULATION)

Choose the correct answer.

A project has produced the following initial draft sequence diagram fragment:



ANSWER: See the explanation for the answer

Explanation:

Correct answer: The message styles for starting the software and reporting that it is ready must be reversed (and, if the session is created by the start action, the first message is a creation message).

`Start Software` is a request from `:User` to `:SoftwareSession`; draw it as a **solid call/create message**, not as a dashed reply.
`Software ready` is the response from `:SoftwareSession` to `:User`; draw it as a **dashed reply message**, not as a solid call.
The `loop` contains the repeated `SoftwareCommand` request and its `ResultDisplay` reply. The subsequent `exit` message may terminate `:SoftwareSession`, shown by the destruction `X`.

If starting the software creates the session, the `Start Software` solid message should terminate at the `:SoftwareSession` object head, with that lifeline beginning at that creation occurrence.

QUESTION NO: 11

Choose the correct answer.

A system engineer specializing in rigid body dynamics wants to model the laws of motion (LOM) so that they may be reused to analyze different types of systems. The LOM consist of the mathematical relations R1 and R2. shown here.

(R1)s=ut + (1/2)at2

(R2) v = u + at

How should the system engineer represent and relate LOM. R1, and R2 in SysML?

A. Create a block for each of LOM. R1 and R? Then create part properties in LOM that are typed by R1 and R?

B. Create an activity for each of LOM. R1, and R2. Then, create actions in LOM that are typed by R1 and R2.

C. Create a constraint block for each of LOM. R1. and R2 Then, create part properties in LOM that are typed by R1 and R2.

D. Create a constraint block for each of LOM. R1, and R2- Then, create constraint properties m LOM that are typed by R1 and R2.

ANSWER: D

Explanation:

Create a constraint block for each of LOM. R1, and R2- Then, create constraint properties m LOM that are typed by R1 and R2. This uses SysML's parametric-modeling mechanism as intended. A constraint block defines a reusable mathematical or logical relation independently of any particular system context. Thus, separate constraint blocks can define the equations for displacement and velocity, including their parameters such as initial velocity, acceleration, time, displacement, and final velocity. The LOM constraint block acts as a higher-level reusable definition that owns constraint properties typed by the R1 and R2 constraint blocks. Each constraint property is a usage of its typed constraint block and permits the individual relations to be composed into the overall laws-of-motion model. When applying the laws to a particular system, a parametric diagram can bind the constraint parameters to value properties of that system using binding connectors. This cleanly separates the reusable equations from the values and structural elements of any vehicle, mechanism, or other analyzed system. SysML specifies constraint blocks and constraint properties as the core elements for expressing such reusable constraints; see the OMG SysML 1.6 specification and the OMG SysML 1.7 specification.